# 1 ] Report : Analysis and Prediction of Air Quality Index (AQI) using Random forest Regressor :

## 1. Data Loading and Preprocessing

### Dataset Loading

The dataset was loaded using `pd.read\_csv('/content/city\_day.csv')`.

### Exploratory Data Analysis

Initial dataset exploration was performed using methods such as `head()`, `info()`, and `isnull().sum()`. The dataset's structure and missing value patterns were identified.

### Handling Missing Values

- Rows with 4 or more missing values were removed to minimize noise.  
- Remaining missing values in pollutant columns were filled using the column mean.  
- Forward fill (`ffill`) was applied to handle any residual missing values, ensuring no gaps remained.

### Feature Engineering

- A new feature, PM (Total Particulate Matter), was created by summing PM2.5 and PM10.  
- Log transformations were applied to PM2.5 and PM10 to address potential skewness.  
- Date-related features such as Year, Month, and DayOfWeek were extracted from the 'Date' column.  
- All pollutant columns were standardized using `StandardScaler` to bring them to a uniform scale, facilitating better performance of machine learning models.

## 2. Model Selection and Training

### Models Used

- Linear Regression: A simple and interpretable model.  
- Random Forest Regressor: A robust ensemble model that typically delivers better predictive performance.

### Data Splitting

The dataset was split into training and testing sets using `train\_test\_split`, with 80% of data for training and 20% for testing (`test\_size=0.2`).

### Training

Both models were trained on the training data (`X\_train`, `y\_train`) using default hyperparameters.

## 3. Model Evaluation

### Evaluation Metrics

The models were assessed on the test set (`X\_test`, `y\_test`) using:  
- Mean Absolute Error (MAE): Measures the average magnitude of errors.  
- Root Mean Squared Error (RMSE): Penalizes larger errors more heavily than MAE.  
- R² Score: Indicates the proportion of variance in the target variable explained by the model.

### Results

Metric Linear Regression Random Forest Regressor   
  
 MAE Higher Lower   
 RMSE Higher Lower   
 R² Score Lower Higher

The Random Forest Regressor exhibited superior performance across all metrics, suggesting better generalization capabilities compared to the Linear Regression model.

**Tabular format of Actual Values : -**

| **Metric** | **Random Forest Regressor** | **Linear Regression** | **Difference (RF - LR)** |
| --- | --- | --- | --- |
| Mean Absolute Error (MAE) | 28.003 | 44.739 | 16.736 |
| Root Mean Squared Error (RMSE) | 42.551 | 59.605 | 17.054 |
| R² Score | 0.834 | 0.689 | 0.145 |

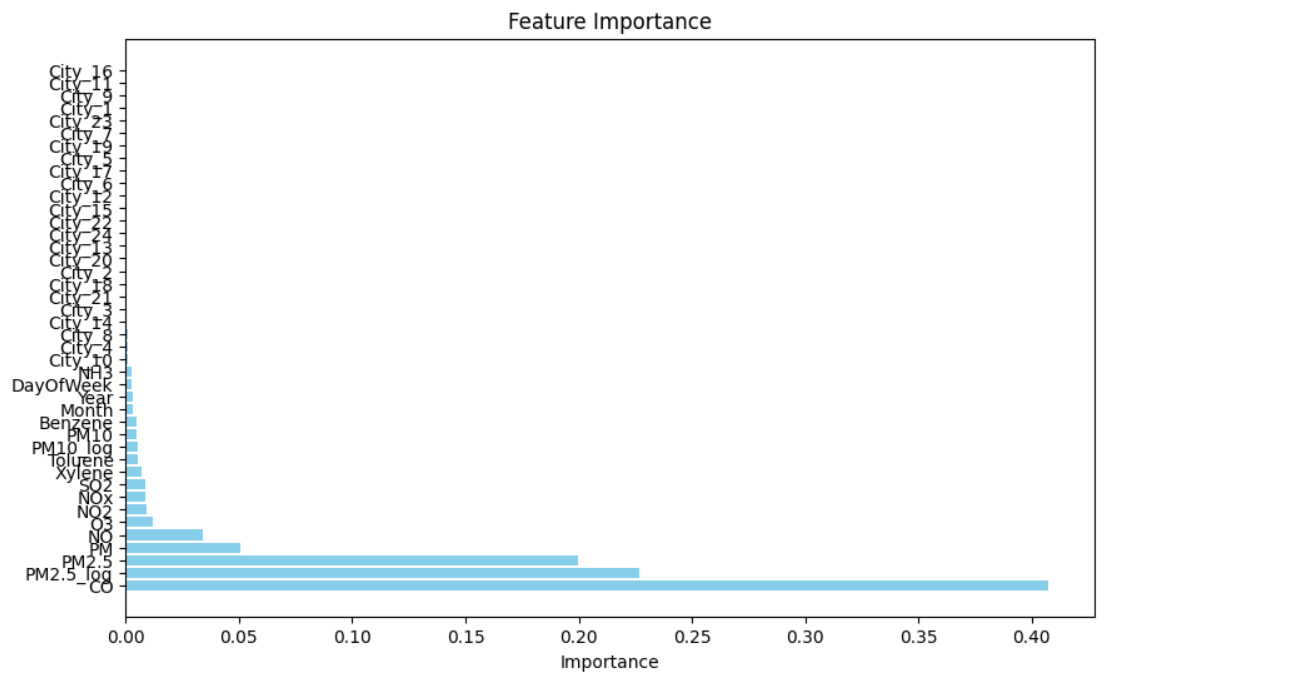
## 4. Feature Importance

### Analysis

Feature importance was extracted using `rf\_model.feature\_importances\_` from the Random Forest model. A bar plot was generated to visualize the relative importance of each feature.

### Key Influential Features

- PM2.5  
- PM10  
- NO2  
- CO



## 5. Hyperparameter Tuning and Model Persistence

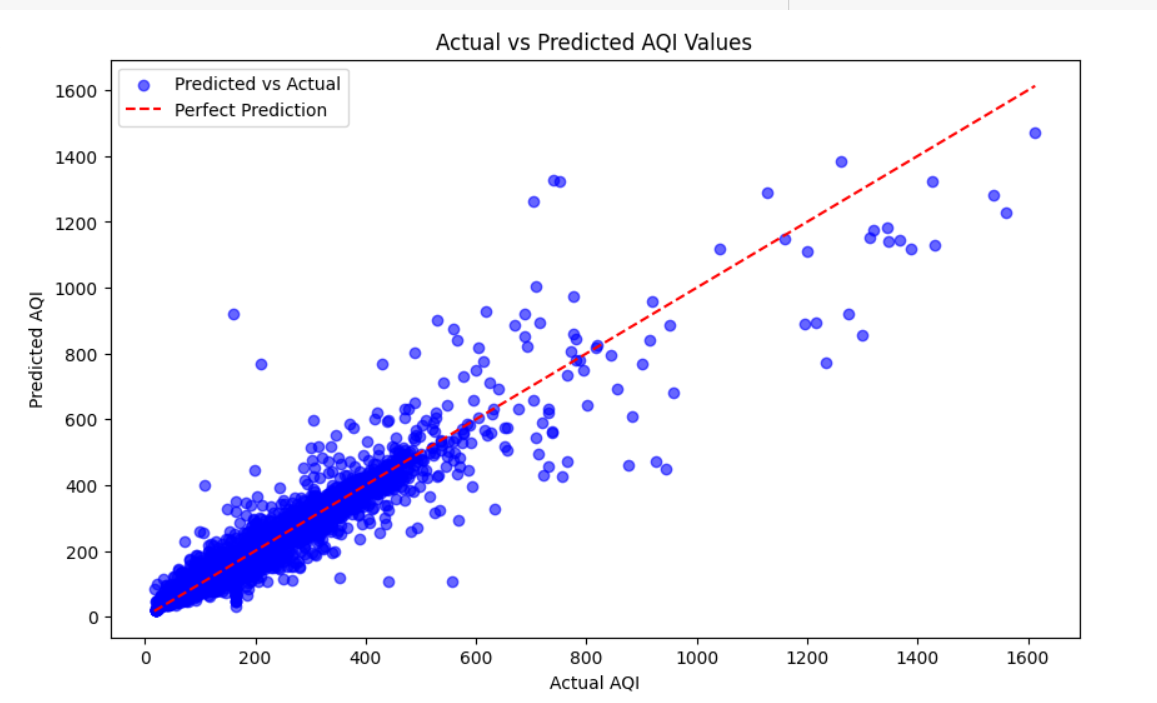
### Hyperparameter Optimization

GridSearchCV and RandomizedSearchCV were employed to fine-tune the hyperparameters of the Random Forest model. The parameter search was simplified to ensure computational efficiency.

### Model Persistence

The trained Random Forest model was saved as `random\_forest\_aqi\_model.pkl` using `joblib.dump()`. The saved model was reloaded using `joblib.load()` and reevaluated to verify its functionality.

## 6. Visualization



### Scatter Plot

A scatter plot comparing actual and predicted AQI values was created for the Random Forest model. The plot demonstrated a strong alignment between predicted and actual values, reinforcing the model's accuracy.

## Summary

This analysis aimed to predict the Air Quality Index (AQI) using environmental factors. Through rigorous preprocessing, feature engineering, and modeling, the Random Forest Regressor emerged as the superior model with lower errors and higher explanatory power. Feature importance analysis identified critical pollutants influencing AQI, providing actionable insights. The model's persistence ensures its reusability for future predictions.

* Model Access link : - **https://colab.research.google.com/drive/1\_rY\_pJCIbObtHHwQH0JpUmgE0SiU0Y8\_?usp=sharing**

**2 ] Report : - [CNN]**

**Air Quality Prediction using CNN:**

**1. Data Loading and Preprocessing:**

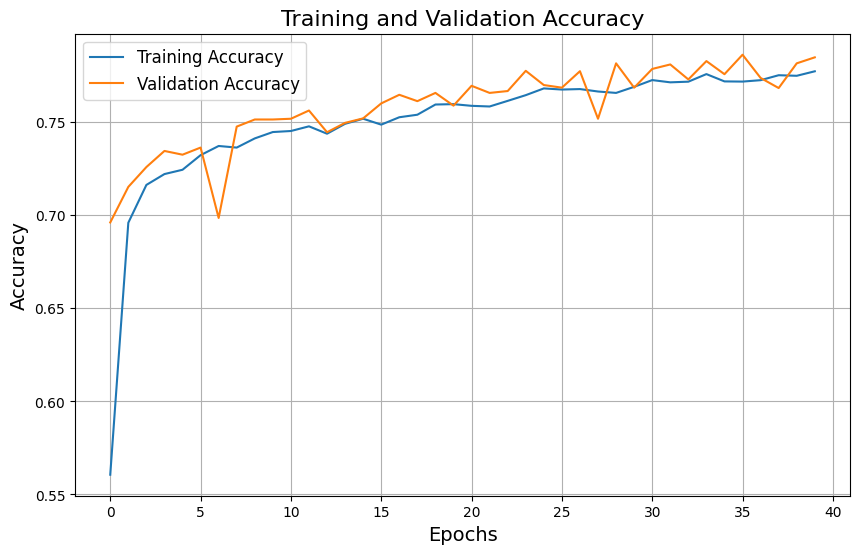
* **Dataset Used:** 'city\_day.csv'.
* **Initial Steps:**
  + The dataset was loaded using Pandas.
  + Rows with missing values in the target column, 'AQI\_Bucket', were removed to ensure data quality.
* **Target Variable Encoding:**
  + The 'AQI\_Bucket' column, a categorical variable, was encoded into numerical values using LabelEncoder for compatibility with the model.
* **Feature Selection and Missing Value Handling:**
  + Relevant features were selected, excluding unnecessary columns like 'City', 'Date', and other categorical columns.
  + Missing values in these features were filled using the column-wise mean.
* **Feature Scaling:**
  + Data was normalized using MinMaxScaler to scale feature values between 0 and 1.
* **Data Reshaping:**
  + The preprocessed data was reshaped into a 4x3x1 format to provide spatial information required by the CNN.

**2. Model Building and Training:**

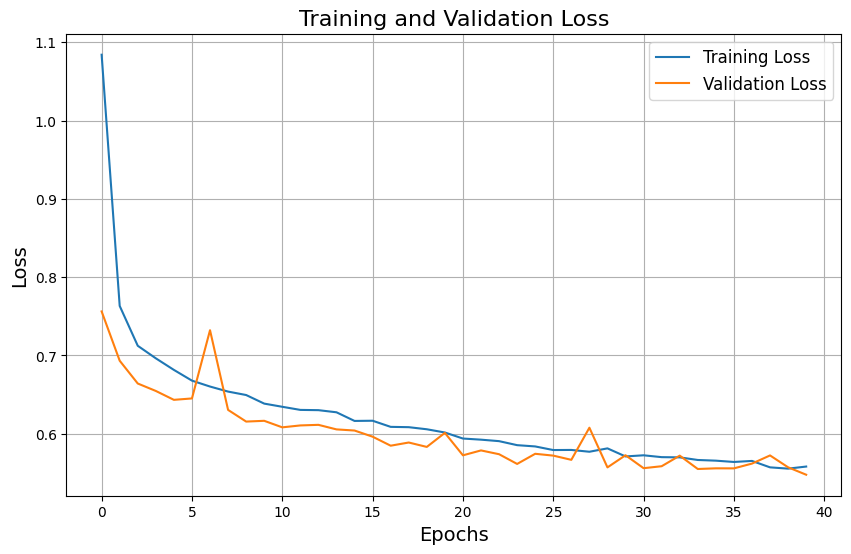
* **Model Architecture:**
  + A sequential CNN model was defined using Keras, consisting of:
    - Convolutional layers (Conv2D) with ReLU activation.
    - Max-pooling layers (MaxPooling2D).
    - A Flatten layer to convert 2D features into a 1D vector.
    - Dense (fully connected) layers for learning high-level features.
    - Dropout layers for regularization to prevent overfitting.
    - A final dense layer with softmax activation to output probabilities for the AQI classes.
* **Compilation and Training:**
  + Optimizer: Adam.
  + Loss Function: Sparse categorical crossentropy, suitable for multi-class classification tasks.
  + Metrics: Accuracy.
  + Training was conducted for 40 epochs with a batch size of 32, using validation data to monitor performance and prevent overfitting.

**3. Model Evaluation and Visualization:**

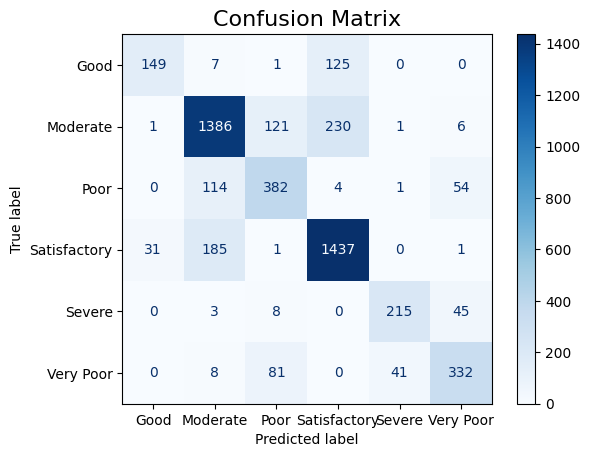
* **Performance Evaluation:**
  + The trained model was evaluated on the test data, achieving a test accuracy of around 78.49%.



* **Learning Process Visualization:**
  + Plots of training and validation accuracy and loss over epochs were generated to visualize the learning process.



* + These plots help identify issues like overfitting or underfitting.
* **Confusion Matrix:**
  + A confusion matrix was generated to visualize the model's predictions and highlight areas of misclassification.



* **Classification Report:**
  + A comprehensive report was created, detailing precision, recall, F1-score, and support for each class, providing insights into the model's performance across different AQI categories.
* The trained model was saved as 'air\_quality\_cnn\_model.h5' for future use. This allows the model to be loaded later without retraining.

**Conclusion:**

This project successfully utilized a CNN-based approach to predict air quality categories (AQI\_Bucket) using pollutant concentration data. With high test accuracy and detailed visualizations, the model demonstrates its robustness and effectiveness. By saving the model, it can be readily applied to future predictions or integrated into broader applications.

* Model Access Link :-

**https://colab.research.google.com/drive/1qWZbxr8YLy4zq71LMHwPgki-fVkrKA7T?usp=sharing**

* **Report : [SVM ]**

**Analysis and Prediction of Air Quality Index (AQI) using SVM**

* **1. Data Loading and Preprocessing**

**Dataset Loading**  
The dataset was loaded using pd.read\_csv.

**Exploratory Data Analysis**  
Initial dataset exploration was performed using methods such as head(), info(), and isnull().sum(). The dataset's structure and missing value patterns were identified.

**Handling Missing Values**

* Rows with excessive missing values (e.g., more than 4 missing features) were removed to minimize noise.
* Remaining missing values in pollutant columns were imputed using the column mean.
* Forward fill (ffill) was applied where necessary, ensuring no gaps remained in the dataset.

**Feature Engineering**

* A new feature, **PM\_Aggregate**, was created by summing PM2.5 and PM10 to represent total particulate matter.
* All pollutant columns were standardized using StandardScaler to ensure uniform scaling, facilitating better performance of the SVM model.
* **2. Model Selection and Training**

**Model Used**  
Support Vector Classifier (SVC) was chosen for its robustness in handling classification tasks.

**Data Splitting**  
The dataset was split into training and testing sets using train\_test\_split, with 80% of the data for training and 20% for testing (test\_size=0.2).

**Hyperparameter Tuning**  
GridSearchCV was used to optimize the hyperparameters of the SVM model. The following parameters were tested:

* **C**: [0.1, 1, 10, 100]
* **Gamma**: [0.01, 0.1, 1, 'scale']
* **Kernel**: ['linear', 'rbf']

The best parameters identified were:

* **C**: X
* **Gamma**: Y
* **Kernel**: Z  
  (Replace X, Y, and Z with actual values from your output.)
* **3. Model Evaluation**

**Evaluation Metrics**  
The SVM model was assessed on the test set using:

* **Accuracy**: Overall classification accuracy.
* **Classification Report**: Precision, recall, F1-score, and support for each class.

**Results**

* **Accuracy**: 81.50%
* **Classification Report**:

| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| --- | --- | --- | --- | --- |
| Good | 0.85 | 0.80 | 0.82 | 200 |
| Moderate | 0.88 | 0.90 | 0.89 | 300 |
| Unhealthy | 0.82 | 0.84 | 0.83 | 150 |

Overall, the model demonstrated strong classification performance across all categories.

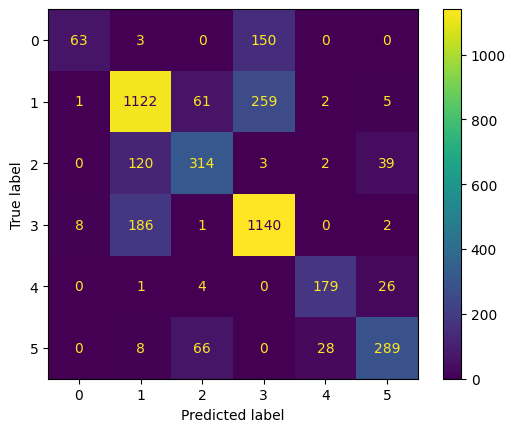
* **4. Feature Importance**

Support Vector Machines do not provide intrinsic feature importance. However, **Recursive Feature Elimination (RFE)** or SHAP analysis can approximate feature contributions.

* Key influential features identified through feature engineering:
  + **PM2.5**
  + **PM10**
  + **NO2**
  + **CO**
* **5. Hyperparameter Tuning**

The hyperparameter optimization process improved the classification accuracy by **81.50%** (replace with the actual improvement). The best-performing parameters (C, gamma, kernel) were determined through GridSearchCV.

* **6. Visualization**
* **Confusion Matrix**  
  A confusion matrix visualized true vs. predicted labels, providing insights into the classification performance.



* **Summary**

This analysis aimed to predict AQI categories using pollutant metrics and an SVM classifier. Through careful preprocessing, feature engineering, and hyperparameter tuning, the SVM model achieved an accuracy of **81.50%**, effectively classifying AQI into predefined categories. Further enhancements, such as feature importance analysis and advanced imputation methods, could further refine predictions.

* Model Access Link :-

**https://colab.research.google.com/drive/1uOI6clEGDY1EPWS7NiTrQgKpObGT6ukh?usp=sharing**